Compression progressive de modèles de plantes à base de cylindres généralisés

S. Mondet[1], F. Boudon[2a], J.C. Hoelt[1], G. Morin[1], R. Grigoras[1], C. Pradal[2b], M. Paulin[1]

[1]Institut de Recherche en Informatique de Toulouse, Université de Toulouse
[2] [a]INRIA, [b]CIRAD, Virtual Plants Team, UMR DAP, TA A-96/02, Avenue Agropolis, 34398 Montpellier Cedex 5, France

Abstract/Résumé

*This paper presents our recent work on progressive compression of plant models based on generalized cylinders. This multi-scale representation is compatible with a direct-acyclic graph representation that allows us to build upon the progressive streaming work of [COM*07]. We present a method for differential coding of plants : an average branch is computed for any chosen group of branches and then, for each branch, we only need to code a transformation and differences. To be able to stream, we identify and take advantage of two types of dependencies : topological (between mother and daughter branches) and dependencies due to differential coding. We obtain a progressive model that makes it possible to select a lightweight representation of a plant while preserving branch density.*

*Ce papier présente nos travaux récents sur la compression progressive de modèles de plantes à base de cylindres généralisés. Cette représentation multi-resolution est compatible avec une représentation sous forme de graphe orienté sans cycle, ce qui nous permet de bénéficier des techniques de streaming progressif proposées dans [COM*07]. Un codage différentiel des plantes est présenté : pour un groupe choisi de branches on calcule une branche moyenne, et pour chaque branche, il reste à coder une transformation et des différences. En vue du* streaming*, nous identifions et exploitons deux types de dépendances : topologiques (entre branche fille et mère) et dues au codage différentiel. Nous obtenons un modèle progressif qui permet la sélection d'une représentation plus légère de la plante, tout en gardant la même densité de branches.*

1. Introduction

Les dernières avancées des technologies 3D (scanner, reconstruction, etc.) et des réseaux ouvrent de multiples perspectives en terme d'applications virtuelles, partagées et/ou collaboratives : exploration de scènes 3D, visites virtuelles, aménagement de paysages etc. Par exemple, le projet Michelangelo de l'Université de Stanford, qui a pour but l'archivage 3D du patrimoine, s'est intéressé à la représentation 3D de statues et a proposé un logiciel, ScanView, pour permettre aux utilisateurs un accès à distance à ces statues. Second Life, un monde virtuel partagé très populaire, encourage ses utilisateurs à créer et échanger des objets 3D. Si actuellement Second Life ne permet que la création d'objets 3D à géométrie solide, les utilisateurs demandent davantage de flexibilité avec l'intégration d'objets plus complexes.

Dans ce contexte, la simulation de scènes naturelles virtuelles est un challenge important pour la représentation de mondes virtuels à des fins scientifiques (architecture, aménagement du territoire, modélisation écologique) ou plus ludiques (jeux vidéo, cinéma).

Le contexte applicatif de ce travail est donc de permettre la mise en ligne de vastes scènes 3D naturelles et réalistes. Une navigation à travers cette scène est offerte à un client à distance. Un téléchargement complet de la scène doit être évité pour diminuer la latence du téléchargement et permettre la navigation à partir d'un client relativement léger. L'idée est de transmettre la scène progressivement, et de

pouvoir visualiser les objets au fur et à mesure de leur réception. L'ordre des objets est décidé relativement à la navigation dans la scène (par exemple, adaptation au point de vue). Certains objets, notamment des plantes, peuvent être volumineux (plusieurs dizaines de Mo), et nous souhaitons donc pouvoir, au niveau d'un seul objet, avoir une représentation progressive. Une telle représentation permet de commencer à visualiser une version simplifiée de l'objet, en attendant que la totalité du modèle soit parvenue au client. Cette représentation progressive, comme les représentations progressives de vidéo, introduisent de la dépendance entre les données. Une donnée ne peut être décodée que lorsque les données dont elle dépend ont déjà été reçues. Par exemple, dans le contexte de vidéo progressive de type MPEG, une image ne peut être décodée que quand l'image de référence dont elle dépend est reçue. Cependant, les données vidéo (ou audio) sont par nature très différentes des données 3D. En effet, les données vidéo ont des contraintes temporelles très fortes : les résultats sur le *streaming* vidéo ne sont donc pas applicables à notre problématique. Une méthode de *streaming* de contenu 3D a été proposée dans [COM*07]. Nous souhaitons utiliser cette méthode pour le streaming de plantes : en proposant un organisation des données compatible avec cette méthode, la mise en paquet, et l'ordonnancement de ces paquets sont déterminés. Pour cela, une représentation progressive adaptée aux plantes et cohérente pour l'approche de *streaming* est proposée. Le modèle original est une représentation d'un arbre par cylindres généralisés. Dans ce papier, un codage différentiel de cette plante est développé duquel découle une représentation progressive.

2. Contexte du travail

2.1. Le projet Natsim

Ce travail s'inscrit dans le cadre du projet ANR *NatSim* [nat05]. Ce projet concerne la définition de structures de données, d'algorithmes et de métaphores pour la simulation, la modélisation, la visualisation et la transmission de scènes naturelles. Afin de permettre une gestion efficace de la complexité inhérente aux scènes naturelles, le coeur du projet repose sur un ensemble de représentations multi-modèles, multi-échelles et multi-résolutions des composants d'une scène naturelle. Ce projet collaboratif et pluridisciplinaire aborde la problématique de la simulation de scènes naturelles à travers cinq *aspects* :

- une structure de données compacte, souple et extensible ;
- les méthodes et processus pour l'acquisition, l'édition et la modélisation ;
- le rendu rendu temps-réel ;
- l'animation et la simulation ;
- l'échange de données par streaming adaptatif.

2.2. Plateforme expérimentale

Afin de permettre l'expérimentation et la recherche dans le domaine de la modélisation par esquisse, du rendu temps-réel, de la simulation de paysage ainsi que des méthodes de travail collaboratif distant en environnement hétérogène, l'IRIT a développé une plateforme expérimentale de visualisation de contenu 3D. L'architecture logicielle ouverte et extensible de cette plateforme expérimentale a été définie afin que les différents partenaires du projet puissent y intégrer leurs travaux en toute simplicité. Ainsi, nos travaux ont été menés afin de rajouter à cette plateforme la possibilité de naviguer à distance dans la scène naturelle.

2.3. Intégration du streaming

Le scénario d'utilisation envisagé pour valider l'apport de notre méthode de mise en paquet pour le streaming de scènes 3D sera le suivant :

Un serveur héberge une scène 3D complexe. Un client/utilisateur à distance souhaite naviguer interactivement dans la scène. Le téléchargement complet du contenu étant hors de propos (contenu beaucoup trop volumineux - pouvant être compté en centaines de Go), le serveur se doit d'adapter (et donc de sélectionner) le contenu au point de vue de l'utilisateur ainsi qu'à l'état de ses ressources (e.g. bande passante) et de le transmettre de manière progressive afin que le client puisse visualiser une représentation (même grossière) de la scène le plus vite possible.

Pour cela nous avons développé la base d'un serveur et d'un client fondés sur la plateforme *NatSim*, en respectant son architecture modulaire.

Le présent travail se concentre particulièrement sur l'optimisation de la transmission progressive des principaux objets des scènes naturelles : les plantes.

3. Travail proposé

Nous nous intéressons donc à la transmission progressive d'objets 3D compressés pour une scène naturelle, plus précisément des plantes. Pour des objets 3D « classiques » il existe plusieurs algorithmes de compression multi-résolutions, que ceux-ci soient représentés par des maillages (e.g. *Progressive Meshes* [KLK04]) ou par des points (c.f. [KB04]).

Cependant, une représentation de plantes par maillages progressifs ne donne pas des résultats satisfaisants [RCB*02] : la topologie d'une plante est telle qu'il est difficile de supprimer des triangles à partir d'un certain niveau. Cette architecture particulière implique d'utiliser des représentations adaptées aux plantes.

Les travaux proposés ici s'appuient sur des représentations par cylindres généralisés [Blo85]. Un premier travail de simplification de cette représentation a été proposé

dans [Bou04], fondé d'une part sur l'importance de chaque branche, et sur la simplification de chaque cylindre généralisé d'autre part.

Ainsi, nous nous proposons de rechercher une méthode de compression progressive de modèles de plantes représentés par des cylindres généralisés et d'exprimer les dépendances internes à ces données afin de les transmettre efficacement. Dans le reste de cet article, nous présentons d'abord brièvement un modèle de mise en paquets (section 4), puis nous explicitons dans les sections suivantes la représentation des plantes proposée.

4. Mise en paquets pour le streaming 3D

La mise en paquets pour le streaming 3D a été assez peu étudiée. Par contre, beaucoup de travaux se sont intéressés à la compression 3D et aux représentations échelonables. Un *état de l'art* a été rédigé par Taubin [Tau99] sur la compression et la transmission progressive de géométrie 3D définie par maillages. Un premier groupe de travaux s'est intéressé à la compression non progressive. Pour la compression progressive, les maillages progressifs sont les plus utilisés. D'autres structures dérivées des maillages progressifs ont été proposées dans *Progressive Forest Split* et *Progressive Simplicial Complexes*.

Les travaux [PKL06, YKK05] minimisent les dépendances entre différentes parties du maillage, de façon à pouvoir afficher une partie de l'objet indépendamment du reste. Gu et Ooi [GO05] ont été les premiers à s'intéresser à la mise en paquets des maillages progressifs. Puis, dans le but de transmettre efficacement des objets 3D multi-résolutions, [COM*07] optimise la qualité visuelle d'un objet partiellement transmis à l'instant t, en considérant à la fois les dépendances entre les données, et leurs importances relatives. Dans ce modèle, considérer les dépendances entre les fragments de données permet de minimiser la présence sur le client de données inutilisables dues aux pertes. Les latences des retransmissions induites par le lien réseau (ici UDP + retransmission) sont prises en compte. Une évaluation statistique avant la transmission tend à maximiser la quantité d'information utilisable pour une bande passante donnée, étant donné les retards dus aux pertes de certains paquets.

Ainsi, l'algorithme de mise en paquets induit par ce modèle permet une visualisation au fil de l'eau. Les données transmises sont visualisées par le client, d'abord avec un faible niveau de détail, puis, au fur et à mesure que de nouvelles données sont disponibles, l'aspect de l'objet visualisé se précise, jusqu'à atteindre sa résolution maximum.

La clé pour pouvoir optimiser le transfert de données est de connaître les dépendances entre les différents niveaux de résolution. Ces dépendances peuvent être modélisées par un graphe orienté sans cycle (*DAG - direct acyclic graph*). De plus, à chaque noeud du graphe correspondant à un fragment des données -ou, à un raffinage de la géométrie- est associé un poids indiquant la contribution de ce fragment au rendu final.

En entrée de l'algorithme de mise en paquets, les objets 3D de la scène doivent avoir une représentation progressive. La représentation utilisée a été celle des maillages progressifs (*progressive meshes* [Hop96]). En effet, il est classique de représenter les objets 3D par des triangulations, et la transformation en maillage progressif est classique et directe. A chaque simplification (*vertex collapse* ou *edge collapse*) la distance du sommet évincé représente une mesure de la contribution de ce sommet à la géométrie. De plus, cette simplification s'appuie sur les éléments -sommets ou arêtes- voisins topologiquement du sommet simplifié. Ces dépendances correspondent donc aux dépendances du *DAG* nécessaire à l'algorithme de mise en paquets et d'ordonnancement.

Néanmoins, à cause de leur topologie ramifiée et leur géométrie particulièrement éparse, les plantes ne se prêtent pas à un codage progressif par maillage : les simplifications sur les triangles peuvent créer des artefacts (c.f. [BMG06]). Ici, nous souhaitons donc proposer une représentation progressive adaptée aux plantes qui puissent satisfaire les deux contraintes en entrée de l'algorithme de *streaming* : les dépendances entre données doivent être modélisées par un *DAG*, et un poids doit être associé à chaque noeud.

5. Représentation progressive des plantes

5.1. Représentation comme cylindres généralisés

Dans ce travail, les plantes considérées « sans leurs feuilles » sont représentées par un ensemble connecté de cylindres généralisés [Blo85]. Dans cette première version de notre travail, nous considérons que les courbes axiales générant les cylindres sont des courbes de Bézier de degré d, et que la fonction rayon est, elle aussi, une fonction de même degré. L'ensemble des cylindres généralisés est organisé sous forme d'une structure de données arborescente n-aire. Nous utilisons par la suite le terme de *n-arbre* pour la structure de données, de façon à éviter les confusions avec l'objet concret « arbre » représenté. Dans cette structure, la racine du *n-arbre* est le tronc de la plante et les branches portées par le tronc sont des filles de ce tronc.

Chaque branche fille contient une position normalisée entre $[0, 1]$ sur sa branche porteuse [PMKL01]. Ce paramètre d'attache u sur la courbe de la branche mère définie le premier point de contrôle de la courbe de Bézier de la branche fille. Les d points de contrôle restants sont codés dans la branche fille par leur trois coordonnées dans l'espace. (c.f. 5).

5.2. L'algorithme de compression des plantes

5.2.1. Codage différentiel d'une plante

Pour optimiser la compression, nous voulons exploiter la similarité des branches. L'idée de l'algorithme de compression est de remplacer le codage absolu d'une branche -les points de contrôle et poids qui la définissent- par une différence par rapport à une courbe de Bézier moyenne, pour un ensemble de branches données. Plus les branches seront similaires, plus il sera possible de quantifier cette différence sur peu de bits et donc de proposer un codage compact. Une première étape consiste donc à regrouper les branches en fonction de leur similarité.

Ainsi, les branches sont d'abord partitionnées en groupe de telle manière que la variabilité géométrique inter-groupe soit minimale pour permettre une quantification avec une perte minimum. Une première solution consiste à utiliser pour cela des algorithmes de partitionnement basés sur des comparaisons inter-branches prenant en compte le nombre de points de contrôle et les variations entre points de contrôle. Dans une première version de ce travail, nous avons discriminé les branches en fonction de leur nombre de points de contrôle.

Pour pouvoir comparer et ensuite coder de manière différentielle les branches, il est nécessaire de normaliser les cylindres généralisés les représentant. Une transformation permettant de passer de la forme normalisée à la forme réelle dans l'espace et inversement sera calculée pour chaque branche. Les représentations normalisées seront utilisées pour calculer les branches moyennes et exprimer les différences de chaque branches à leurs branches moyennes de référence sous forme de delta de points de contrôle. Nous avons choisi de transformer les cylindres afin que le premier point de contrôle P_0 de la courbe de Bézier génératrice coïncide avec l'origine et le dernier P_N avec le point $(0,0,1)$. Cette première transformation définit deux angles de rotation, et un facteur d'échelle (nous avons choisi d'utiliser une mise à l'échelle uniforme). Il reste ensuite un degré de liberté, qui correspond à la rotation autour de l'axe $\vec{z}$. Pour cela, nous choisissons l'orientation autour de l'axe $\vec{z}$ en ramenant le centre de gravité $\overline{P_i}$ des points de contrôle restants dans le plan xz.

Les transformations T résultantes de cette normalisation sont une translation $t = -\overrightarrow{P_0}$, une mise à l'échelle $s = \frac{1}{\|\overrightarrow{P_0P_N}\|}$, et 3 angles de rotation tel que $\overrightarrow{T(P_0)T(P_N)} = \vec{z}$ et $\overrightarrow{T(P_0)T(\overline{P_i})} \cdot \vec{y} = 0$. Les transformations inverses seront conservées pour chaque branche et permettent d'instancier une forme normalisée dans l'espace 3D.

Lorsque toutes les courbes génératrices des branches d'un groupe ont été normalisées, la courbe moyenne est calculée simplement telle que ses points de contrôle $\overline{P_i}$ soient les barycentres des points de contrôle P_i des n courbes.

Pour chaque branche, on pourra ensuite redéfinir les courbes de manière différentielle en fonction des courbes moyennes en stockant les coordonnées des points de contrôle sous la forme de différences aux points moyens. Il est ainsi possible de coder ces différences sur un nombre restreint de bits. On notera que les courbes étant normalisées, le premier et dernier points de contrôle n'ont pas besoin d'être codés. Pour des courbes de Bézier de degré 3 par exemple, seuls les deux points intermédiaires ont besoin d'être définis. Ainsi, le codage d'une branche est maintenant défini par des paramètres d'instanciation (une transformation) et une forme différentielle de courbe normalisée. La translation de la transformation est donnée par un scalaire définissant la position sur la branche porteuse (donnant auparavant le premier point de contrôle), trois scalaires pour les angles de rotation et un scalaire pour le changement d'échelle uniforme. Aussi, en plus du pointeur sur la branche mère qui était déjà présent dans la représentation originale, un pointeur supplémentaire vers le modèle est nécessaire.

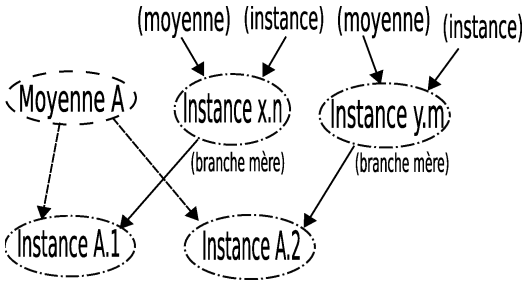

Figure 1: *Un exemple d'un certain niveau dans le DAG : par exemple l'instance notée A.1 dépend du modèle de branche A (codage différentiel) et de l'instance x.n (structure topologique de la plante).*

5.2.2. Expression des dépendances

Pour utiliser le codage différentiel d'une plante comme une représentation progressive de la plante en vue de la transmission pour une visualisation au fil de l'eau, il est nécessaire d'exprimer les dépendances et d'affecter un poids à chaque raffinage. Deux familles de dépendances existent : les premières sont les dépendances dues à la structure topologique (*n-arbre*) de la plante. C'est à dire, la dépendance entre une branche fille et la branche mère à laquelle elle est attachée. La deuxième famille correspond aux dépendances dues au codage différentiel. Une branche dépend de la branche moyenne. La figure 1 montre sur un niveau les deux formes de dépendances.

Pour pouvoir représenter la plante complète sans pour autant connaître toutes les branches en résolution maximum, il est nécessaire qu'une branche fille ne dépende pas de la version 'finale' (contenant tous les détails originaux) de sa mère. Pour cela, nous décomposons une branche donnée en niveaux de détails et déterminons la résolution nécessaire à

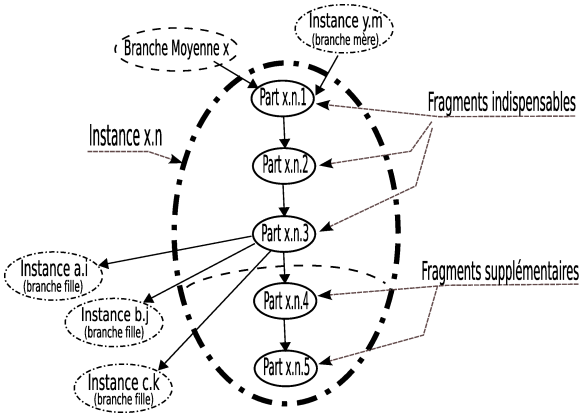

Figure 2: *Un exemple d'instance étendue : le noeud correspondant à une branche est décomposé en sous noeud qui contiennent soit des informations de transformation, soit des informations de raffinage géométrique. De cette façon, on peut exprimer un nombre de niveaux de détail arbitraire pour une branche.*

l'attache d'une branche fille. Par exemple, dans la figure 2 les branches filles de la branche $x.n$ peuvent être visualisées dès que les niveaux de détails $x.n.1$, $x.n.2$, et $x.n.3$ ont été reçus.

Ces dépendances sont forcément non cycliques ; cette structure de données peut donc effectivement être traitée par l'algorithme vu dans la section 4.

5.2.3. Évaluation de l'importance des informations

En plus de l'expression des dépendances, pour pouvoir décider de la mise en paquets et de l'ordonnancement de l'envoi de ces paquets, il faut aussi quantifier la contribution des informations fournies par chaque noeud du *DAG* à la reconstruction de la scène.

Dans notre implémentation, nous avons pour chaque branche une représentation en niveaux de détails (telle que le montre la figure 2) à deux niveaux. Le premier niveau correspond à l'instanciation de la branche moyenne, c'est à dire, codant la transformation complète. Le deuxième niveau correspond à l'introduction des différences à la branche moyenne.

Il parait d'abord naturel de privilégier les paramètres d'instanciation pour qu'au minimum toutes les branches soient représentées comme instance d'une branche moyenne, en omettant au départ le détail contenu dans le codage différentiel. De façon à favoriser d'abord l'introduction de nouvelles branches, nous ne donnons qu'un poids faible aux noeuds stockant les paramètres différentiels par rapport aux noeuds stockant les paramètres d'instanciation mais proportionnel au poids de la branche pour qu'une fois que toutes les branches soient représentées, les branches les plus importantes soient raffinées en premier.

Pour ordonner les branches, nous avons choisi, dans une première version, d'utiliser la structure hiérarchique de la plante. Nous donnons à une branche une importance proportionnelle au nombre de ses branches descendantes. Les branches terminales -feuilles du *n-arbre-* ont par défaut le poids 1, et chaque branche intermédiaire reçoit la somme des poids des branches qu'elle porte.

Finalement pour ordonner les informations des branches de même importance dans la hiérarchie, une idée est prendre en compte leur importance visuelle. Pour cela, une heuristique basée sur le volume des branches est proposée dans [Bou04] et que nous souhaitons intégrer à terme pour raffiner notre méthode de détermination du poids des noeuds.

5.2.4. Résultats

Les résultats sur un premier jeu de données de 3 arbres sont présentés dans le tableau 4. Les deux premières plantes, un pommier et un noyer, sont reconstruites à partir de mesures de digitalisation par des biologistes [CSKG03, SRG97]. Le troisième est un arbre simulé par L-systems [PL90].

Le codage différentiel représente le même arbre que l'arbre original, à la quantification près des coefficients de différence des points de contrôle sur 8 bits. Le modèle compressé sans les différences donne une représentation plus légère (entre 0.54 pour l'arbre généré par L-système, et 0.72 pour le pommier), pour une densité équivalente de branches. Ces résultats sont encourageants car les regroupements et la quantification sont faits de manière relativement naïve. La figure 3 montre un exemple de recontruction du noyer avec et sans les coefficients différentiels.

6. Conclusion et perspectives

Nous avons proposé un codage différentiel pour des plantes représentées par une arborescence de cylindres généralisés. Les premiers résultats sont encourageants. De nombreuses pistes restent néanmoins à explorer.

Dans ce premier travail, les regroupements de branches est fait de manière simple en considérant uniquement le nombre de points de contrôle. Il serait intéressant de prendre en compte la précision et la taille des données issues de la quantification faite sur les points de contrôle pour déterminer les groupes.

Une étude de sensibilité sur l'impact des différents paramètres définissant notre modèle de branches sur la représentation finale peut nous permettre de caractériser leurs importances et ainsi créer de nouveaux niveaux de détails. En effet, certaines valeurs de paramètres peuvent varier très faiblement et avoir un impact minime sur la la représentation.

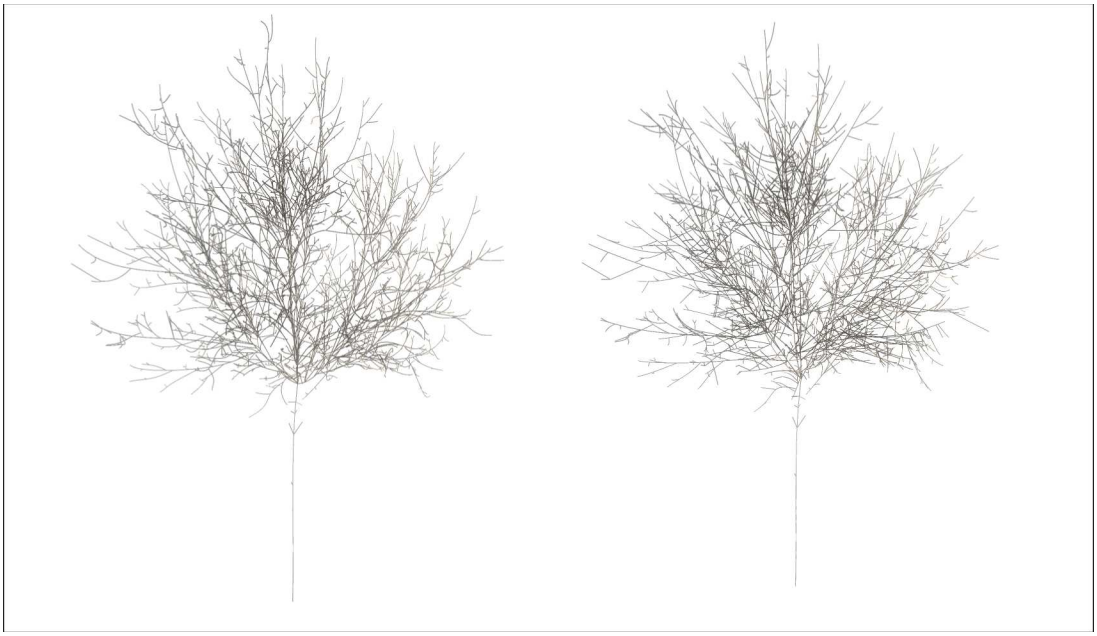

Figure 3: *Exemple de reconstruction d'une plante (Noyer numérisé [SRG97]) avec et sans les coefficients différentiels : à gauche compression sans pertes (74 % du modèle original), à droite, avec pertes (61 % du modèle original).*

Nom	Original	Compression *sans pertes* (ratio)	Sans les différences (ratio)
Pommier	109678	90668 (0.83)	78980 (0.72)
Noyer	580788	429047 (0.74)	353975 (0.61)
L-système	1519978	1028551 (0.68)	817687 (0.54)

Figure 4: *Résultats expérimentaux prennant en compte uniquement la géométrie (courbes de Bézier des branches) et les références nécessaires (références topologiques inter-branches et références aux branches moyennes). Valeurs données* ***en bits*** *(codage des réels : 32 bits, des différences : 8 bits).*

Il paraîtrait donc judicieux de différer la transmission de tels paramètres.

Pour des modèles assez réguliers, des valeurs prédictives, déduites de connaissances biologiques des plantes, pour certains paramètres tels que les angles de branchements peuvent être utilisées en première approximation sans nécessiter de transmission par le réseau.

Nous envisageons d'autre méthodes de compression des branches basées par exemple sur l'analyse en composante principale (ACP) des paramètres d'un groupe de branches permettraient de coder les différences sur une base de représentation des données hiérarchique et donc adaptée à notre problème.

Finalement, des tests sur différents types de réseaux et différents systèmes (Ordinateurs, PDA, etc.) nous permettront en pratique de mieux caractériser l'adaptation de notre méthode au contexte du streaming.

7. Remerciements

Ce travail a été financé par le projet ANR NatSim 05-MMSA-0004-01. Nous remercions C. Godin pour sa contribution au projet et sur les structures multi-échelles, E. Costes et H. Sinoquet pour nous fournir des bases de données de plantes digitalisées.

References

[Blo85] BLOOMENTHAL J. : Modeling the mighty maple. *ACM Computer Graphics (Siggraph'85) 19*, 3 (1985), 305–311.

[BMG06] BOUDON F., MEYER A., GODIN C. : *Survey on Computer Representations of Trees for Realistic and Efficient Rendering*. Tech. rep., LIRIS UMR 5205 CNRS, 2006.

[Bou04] BOUDON F. : *Représentation géométrique multi-échelles de l'architecture des plantes*. PhD thesis, Université de Montpellier II, 2004.

[COM*07] CHENG W., OOI W. T., MONDET S., GRIGORAS R., MORIN G. : An analytical model for progressive mesh streaming. In *MULTIMEDIA '07 : Proceedings of the 15th ACM international conference on Multimedia* (2007), ACM Press.

[CSKG03] COSTES E., SINOQUET H., KELNER J., GODIN C. : Exploring within-tree architectural development of two apple tree cultivars over 6 years. *Annals of Botany 91* (2003), 91–104.

[GO05] GU Y., OOI W. T. : Packetization of 3D Progressive Meshes for Streaming over Lossy Networks. In *Proceedings of the 14th International Conference on Computer Communications and Networks (ICCCN)* (2005).

[Hop96] HOPPE H. : Progressive meshes. In *SIGGRAPH'96 Conference Proceedings* (1996).

[KB04] KOBBELT L., BOTSCH M. : A survey of point-based techniques in computer graphics. *Computers and Graphics 28*, 6 (2004), 801–814.

[KLK04] KIM J., LEE S., KOBBELT L. : View-Dependent Streaming of Progressive Meshes. *Shape Modeling Applications* (2004).

[nat05] The Nature Simulation Project, 2005. ANR 05-MMSA-0004-01 `http://www.irit.fr/NatSim/`.

[PKL06] PARK S.-B., KIM C.-S., LEE S.-U. : Error resilient 3-D mesh compression. *IEEE Transactions on Multimedia 8*, 5 (October 2006), 885–895.

[PL90] PRUSINKIEWICZ P., LINDENMAYER A. : *The algorithmic beauty of plants*. Springer Verlag, 1990.

[PMKL01] PRUSINKIEWICZ P., MÜNDERMANN L., KARWOWSKI R., LANE B. : The use of positional information in the modeling of plants. *ACM Computer Graphics (Siggraph'01) 22*, 4 (2001), 289–300.

[RCB*02] REMOLAR I., CHOVER M., BELMONTE O., RIBELLES J., REBOLLO C. : Geometric simplification of foliage. In *Proc. of the Eurographics02 Short Presentations* (2002), pp. 397–404.

[SRG97] SINOQUET H., RIVET P., GODIN C. : Assessment of the three-dimensional architecture of walnut trees using digitising. *Silva Fennica 31*, 3 (1997), 265–273.

[Tau99] TAUBIN G. : 3D Geometry Compression and Progressive Transmission, 1999.

[YKK05] YAN Z., KUMAR S., KUO C.-C. : Mesh segmentation schemes for error resilient coding of 3-D graphic models. *Circuits and Systems for Video Technology, IEEE Transactions on 15*, 1 (2005), 138–144.

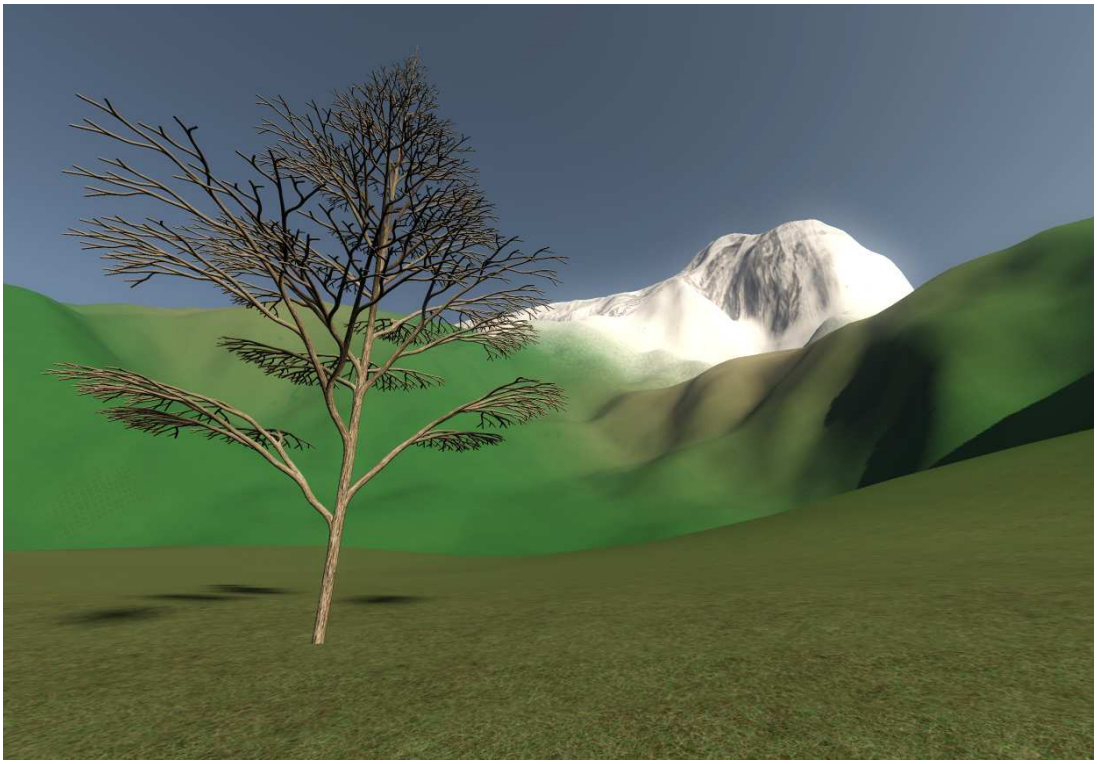

Figure 5: *Exemple d'utilisation de cylindres généralisés pour le rendu de plantes (ici un arbre généré par L-sytème).*